|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ **Информатика и системы управления**

КАФЕДРА **Компьютерные системы и сети (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.03.01 Информатика и вычислительная техника**

**Отчет**

|  |  |
| --- | --- |
| **по лабораторной работе №** | 5 |

**Дисциплина:** Языки интернет-программирования

**Вариант:** 23

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ6-33Б |  |  | Д. И. Мироненко |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | В. Д. Шульман |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2023

**Часть 1**

**Задание**

Вычислить: .

**Код программы**

**Файл “main.rb”**

|  |
| --- |
| # frozen\_string\_literal: true  require\_relative 'utils'  print "Enter 'a': "  a = gets.chomp  print "Enter 'b': "  b = gets.chomp  puts "Result: #{f(a, b)}" |

**Файл “utils.rb”**

|  |
| --- |
| # frozen\_string\_literal: true  def f(param\_a, param\_b)  a = param\_a.to\_f  b = param\_b.to\_f  (Math.sin(a) - b) / (b.abs + Math.cos(b\*\*2))  end |

**Файл “test\_utils.rb”**

|  |
| --- |
| # frozen\_string\_literal: true  require 'minitest/autorun'  require\_relative 'utils'  # Test `utils.rb`  class TestUtils < Minitest::Test  def test\_f  assert\_in\_delta(-1.0, f(5, 9), 0.1)  assert\_in\_delta(-0.9, f(8, 3), 0.1)  assert\_in\_delta(0.8, f(1, 0), 0.1)  end  end |

**Работа программы**

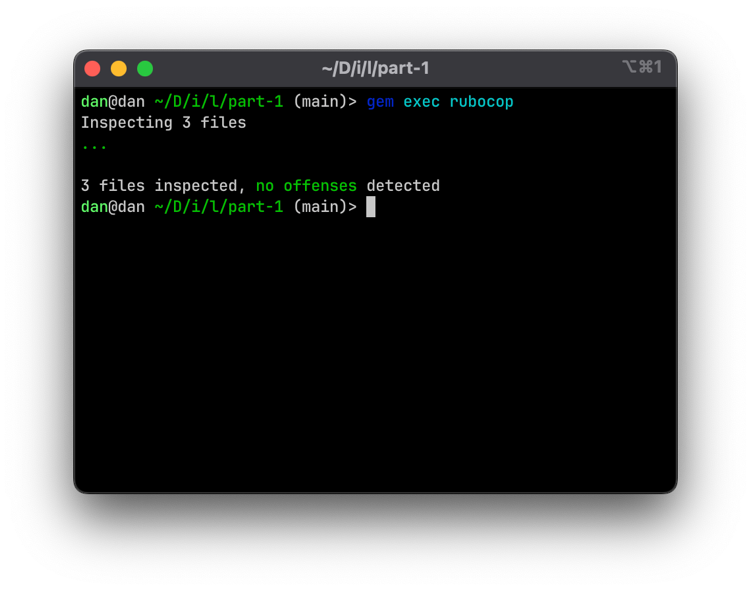


**Рисунок 1** – работа программы

**A screenshot of a computer program

Description automatically generated**

**Рисунок 2** – выполнение unit-тестов

****

**Рисунок 3** – проверка кода rubocop’ом

**Часть 2**

**Задание**

Дана строка, состоящая из дат в ISO-формате, разделённых одним пробелом. Найти самую близкую дату к сегодняшнему дню. (2022-01-11 2022-01-24 . . . )

**Код программы**

**Файл “main.rb”**

|  |
| --- |
| # frozen\_string\_literal: true  require 'date'  require\_relative 'utils'  print 'Enter dates in ISO format: '  dates = gets.chomp.split  today = Date.today  puts "Closest date: #{get\_closest\_date(dates, today)}" |

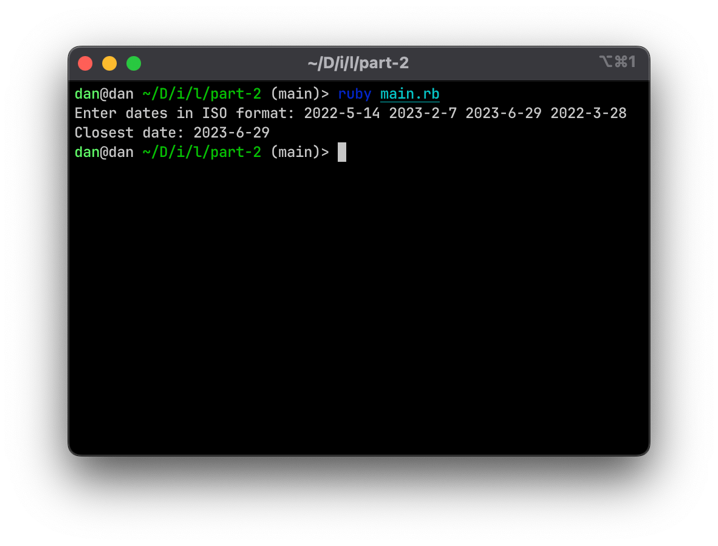
**Файл “utils.rb”**

|  |
| --- |
| # frozen\_string\_literal: true  require 'date'  def get\_closest\_date(dates, target\_date)  dates.min\_by do |date|  (Date.parse(date) - target\_date).abs  end  end |

**Файл “test\_utils.rb”**

|  |
| --- |
| # frozen\_string\_literal: true  require 'minitest/autorun'  require 'date'  require\_relative 'utils'  # Test 'utils.rb'  class TestUtils < Minitest::Test  def test\_get\_closest\_date1  dates = ['2022-5-14', '2023-2-7', '2023-6-29', '2022-3-28']  target\_date = Date.today  assert\_equal '2023-6-29', get\_closest\_date(dates, target\_date)  end  def test\_get\_closest\_date2  dates = ['2022-5-14', '2023-11-4', '2023-6-29']  target\_date = Date.today  assert\_equal '2023-11-4', get\_closest\_date(dates, target\_date)  end  def test\_get\_closest\_date3  dates = ['2023-10-3', '1998-3-5', '1965-2-19']  target\_date = Date.today  assert\_equal '2023-10-3', get\_closest\_date(dates, target\_date)  end  end |

**Работа программы**

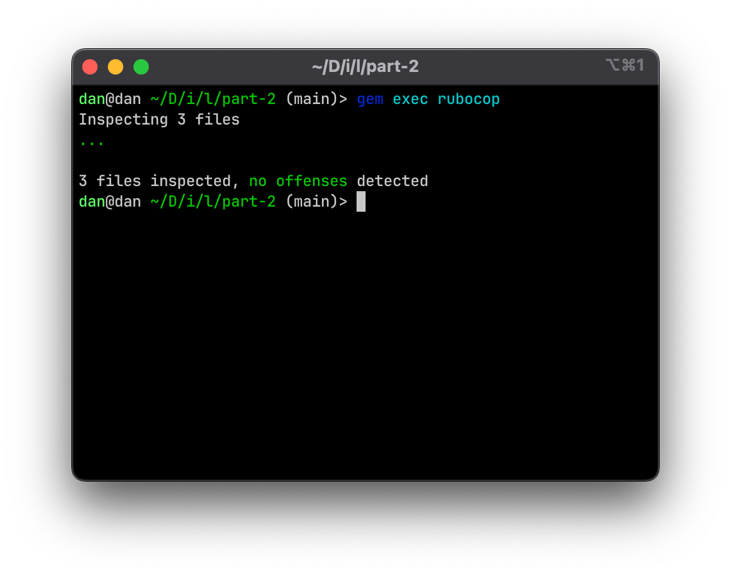


**Рисунок 4** – работа программы

**A screenshot of a computer program

Description automatically generated**

**Рисунок 5** – выполнение unit-тестов

****

**Рисунок 6** – проверка кода rubocop’ом

**Часть 3**

**Задание**

Дана последовательность строк. Каждая строка состоит из слов, разделенных пробелами. Написать программу, обеспечивающую ввод строк и их корректировку. Корректировка заключается в следующем. Если слово содержит символы, отличные от букв латинского алфавита и цифр, то удалить его. Если слово состоит из букв латинского алфавита и цифр и начинается с цифры, заменить эту цифру символом «\_» подчеркивание. Вести подсчет количества корректировок. Вывести на печать исходные и скорректированные последовательности строк. Автоматический тест программы обязательно должен генерировать случайные строки в соответствии с правилами, перечисленными в задании.

**Код программы**

**Файл “main.rb”**

|  |
| --- |
| # frozen\_string\_literal: true  require\_relative 'utils'  SEPARATOR = '-' \* 30  print 'Strings count: '  strings\_count = gets.to\_i  strings = input(strings\_count)  puts SEPARATOR  puts 'Source text:'  show strings  corrections\_count = fix\_text(strings)  puts SEPARATOR  puts "Fixed text (corrections: #{corrections\_count}):"  show strings |

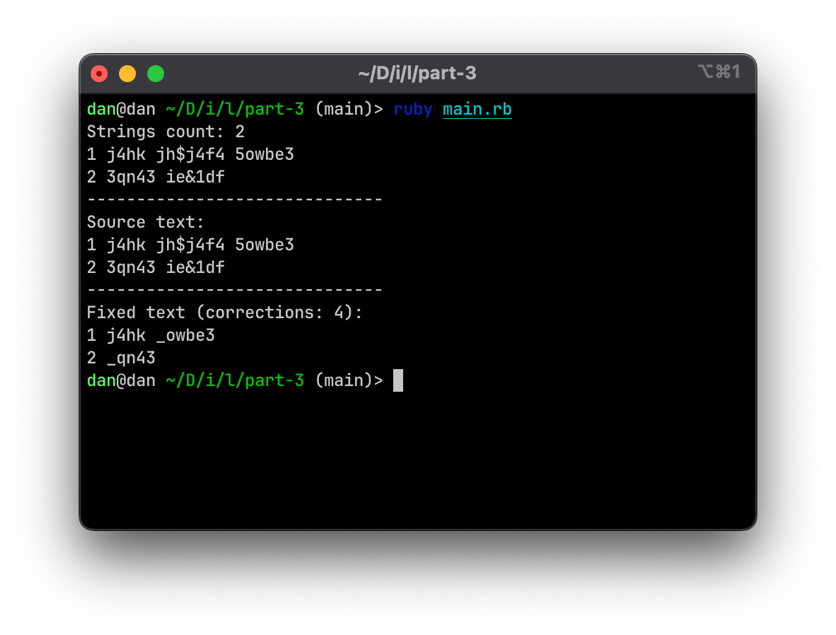
**Файл “utils.rb”**

|  |
| --- |
| # frozen\_string\_literal: true  def alnum?(str)  /^[[:alnum:]]+$/.match?(str)  end  def numeric?(str)  /^[[:digit:]]+$/.match?(str)  end  def fix\_text(strings)  corrections = 0  strings.size.times do |i|  new\_string = []  strings[i].size.times do |j|  if alnum? strings[i][j]  if numeric? strings[i][j][0]  new\_string.push("\_#{strings[i][j][1..]}")  corrections += 1  else  new\_string.push(strings[i][j])  end  else  corrections += 1  end  end  strings[i] = new\_string  end  corrections  end  def input(count)  raise StandardError, "'count' cannot be a negative number!" if count.negative?  strings = []  count.times do |i|  print "#{i + 1} "  strings.push(gets.split)  end  strings  end  def show(strings)  strings.size.times do |i|  puts "#{i + 1} #{strings[i].join(' ')}"  end  end |

**Файл “test\_utils.rb”**

|  |
| --- |
| # frozen\_string\_literal: true  require 'minitest/autorun'  require\_relative 'utils'  # Test 'fix\_text' function  class TestFixText < Minitest::Test  def test\_some\_text  strings = [  %w[j4hk jh$j4f4 5owbe3],  %w[3qn43 ie&1df]  ]  assert\_equal 4, fix\_text(strings)  assert\_equal [%w[j4hk \_owbe3], ['\_qn43']], strings  end  def test\_emtpy\_text  strings = [[]]  assert\_equal 0, fix\_text(strings)  assert\_equal [[]], strings  end  def generate\_random\_word  chars = ('a'..'z').to\_a + ('A'..'Z').to\_a + (0..9).to\_a + ['\_', '!', '@', '#', '$', '%', '^', '&', '\*']  len = rand(15)  word = ''  len.times do  word += chars.sample.to\_s  end  word  end  def generate\_random\_strings  strings = []  rand(15).times do  new\_strings = []  rand(15).times do  new\_strings.push(generate\_random\_word)  end  strings.push(new\_strings)  end  strings  end  end |

**Работа программы**



**Рисунок 7** – работа программы

**A screenshot of a computer

Description automatically generated**

**Рисунок 8** – выполнение unit-тестов

**A screenshot of a computer program

Description automatically generated**

**Рисунок 9** – проверка кода rubocop’ом

**Вывод**

Изучил основы языка программирования ruby, написав несколько программ; познакомился с unit-тестированием; проверил код на наличие ошибок с помощью линтера rubocop.